SAFECode
Security Development Lifecycle (SDL)

Michael Howard – Microsoft
Matthew Coles – EMC

15th Semi-annual Software Assurance Forum,
September 12 - 16, 2011

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Agenda

- Introduction to SAFECode
- Security Training and Advocacy
- Secure Development Practices
- Integrity Controls in the Supply Chain
- Summary / Wrap-up

We enjoy audience participation – please ask your questions at any time, or feel free to share your own stories

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

About SAFECode

The Software Assurance Forum for Excellence in Code (SAFECode) is a global, industry-led effort to identify and promote best practices for developing and delivering more secure and reliable software, hardware and services

Juniper Networks
symantec
Microsoft
NOKIA
SAP
Adobe
EMC²
where information lives®

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

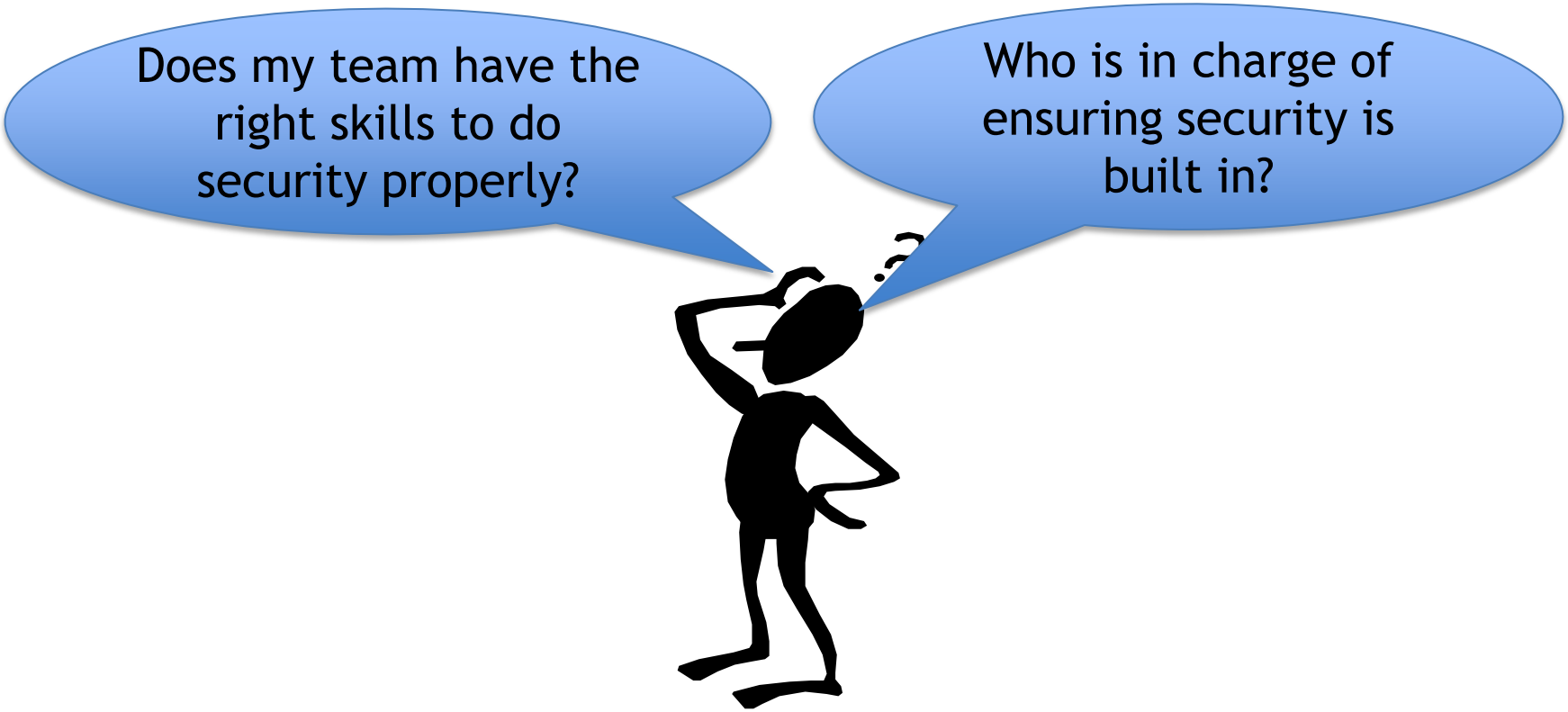
SAFECode's View of Software Assurance

Software Assurance: Confidence that software, hardware and services are free from intentional and unintentional vulnerabilities and that the software functions as intended.

In practice, software vendors take action in three key, overlapping areas to achieve software assurance—security, authenticity and integrity.

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Thoughts to Consider



Are you outsourcing security knowledge?

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Security Training & Advocacy

Organizational structures should be leveraged to maintain security

- Executive Management backing and enforcement

- Core security team
  - Define Policy and Standards
  - Develop key expertise (SMEs)
  - Oversight and direction to the organization

Security Training & Advocacy

Advocates & Champions

- Business level security resources embedded with their peers
  - Champions of security at all levels of the organization
  - Speaking the language of the team
  - Driving best practices locally

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Security Training & Advocacy

Security awareness is not usually taught in school

Prospective Employer: So, you have a computer degree from a prestigious university. Impressive.

Prospective Employer: Tell me what you know about developing software.

Prospective Employee: The Cloud is where it's at. I can write functional J2EE code at 300 lines an hour!

Prospective Employer: Is it secure?

Prospective Employee: Umm… I don't know…

- Team members need to know how to recognize and understand the choices they make affect secure operations of a product or service

Security Training & Advocacy

Classic decision point: Build or Buy

- Buying security expertise training requires finding a good training partner

Software Engineering Institute | Carnegie Mellon

SANS

Safelight

cigital
Software Confidence. Achieved.

PEOPLE SECURITY

SecurityInnovation
THE SOFTWARE SECURITY COMPANY

- Building requires security expertise to exist already…

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Security Training & Advocacy

Consider a phased approach

- **Buy** to ramp up quickly
  - Create security awareness training offerings, to get team members thinking about security
  - Develop a security training curriculum
  - Understand your organization's skills gaps

- **Build** unique "organization specific" training based on the business needs and structure

Security Training & Advocacy

Another classic decision: ILT or CBT

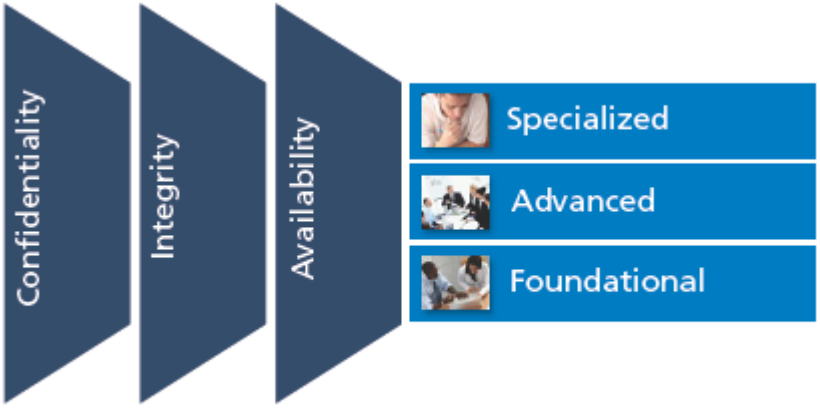
Computer-based	Instructor-led
Flexible schedule	Get direct answers
Cost effective	May accommodate labs
Suited for global orgs	In-house mentoring
Can intermix COTS content	Build peer resources

- Across member companies, we find a hybrid approach works well
  - Bootcamps and hands-on training for very specialized activities or to ramp up quickly
  - Technology-specific CBTs for quick reference

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Security Training & Advocacy

- Curriculum development is important to keep the team engaged and current
- Multiple levels of training, based on needs and roles



Three levels of security engineering training

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Security Training & Advocacy

Plan ahead

- Suggest business leaders take goals to have their team trained on security
- Team members should look ahead to what skills they need for a particular project
- Take training between releases

Keep training records

- Make sure key individuals have the skills they need to manage security during the lifecycle
- Keep the skills of the team current as threats evolve

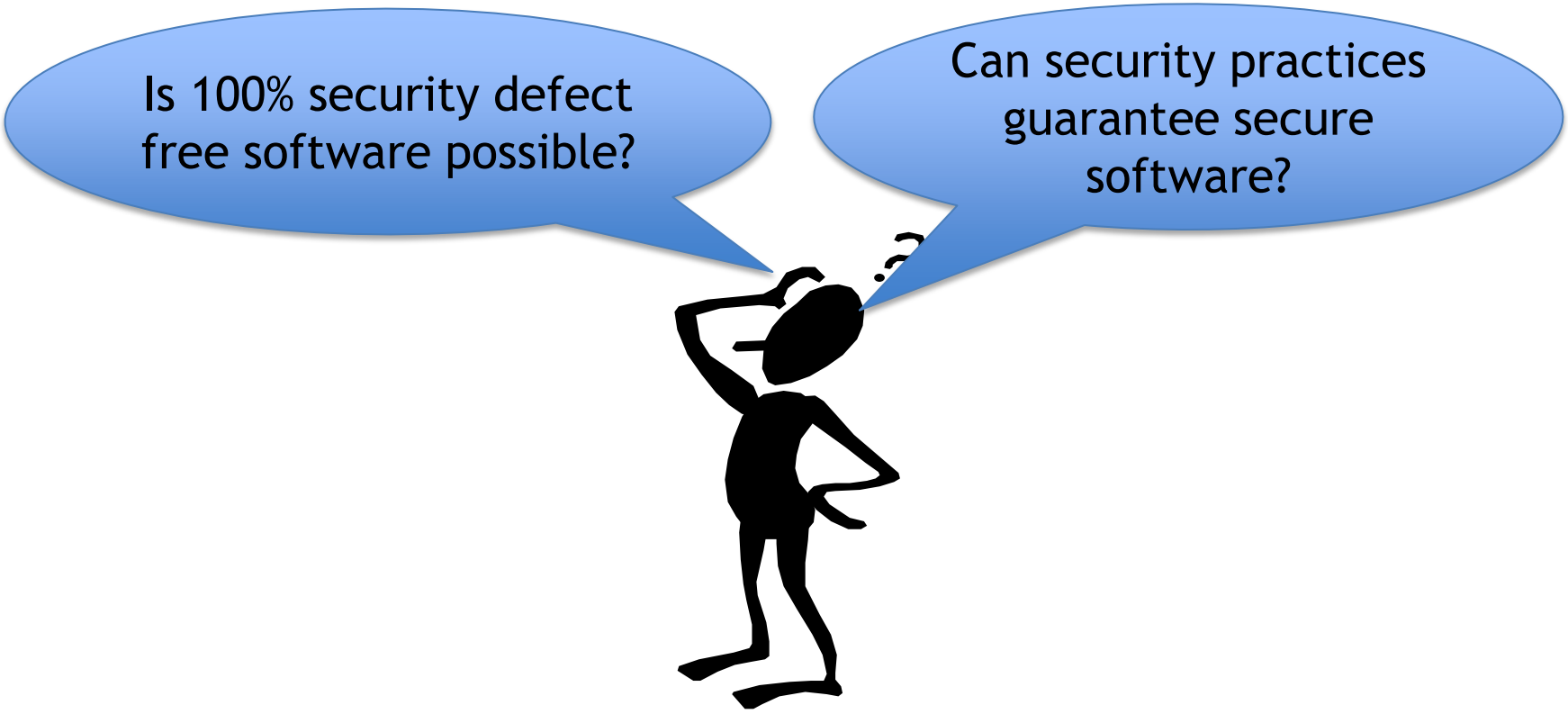
SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Security Training & Advocacy

- Everyone needs to be security **aware**.
- Key individuals need to be security **focused**.
- Enable individuals at all levels to become more knowledgeable.
- Empower certain individuals to have more direct control over security

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Thoughts to Consider



Is your goal to know how much you don't know?

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Secure Development

The goal of the paper is to describe practices that:

- Are in use by SAFECode members
- Known to improve software security

Secure Development

History

- Original paper was published Oct 2008
- Current paper was published Feb 2011
- Most technical material overhauled
- Added CWE references (45+ base and parent weaknesses)
- Added Verification sections
- Removed training and code handling sections

Secure Development

Secure Design – Threat Modeling

- A way of identifying risks to the software before it is built
- Many methods available
  - STRIDE
  - Risk Analysis
  - Misuse Cases
  - Threat Library
  - Brainstorming (no SAFECode members do this!)
- Scoring risks varies widely, too

Secure Development

Secure Design – Use Least Privilege

- Specific guidance available on how to achieve this goal for various OSes
- A very important way to help contain or prevent damage

Secure Development

Secure Design - Implement Sandboxing

- A more aggressive least privilege mechanism
- Attempts to isolate a process to only a small set of resources (memory, files etc)
- Newer languages do this by design
  - C#, Java etc
- Can be a harder problem for C and C++ processes
- Should be used by very high exposure apps
  - Acrobat, IE, Chrome etc

Coding - Minimize Use of Unsafe String & Buffer Functions

Secure Development

- Consider alternate secure functions instead (sampling)

Unsafe Function	Safer Function
strcpy	strcpy_s
strncpy	strncpy_s
strcat	strcat_s
strncat	strncat_s
scanf	scanf_s
sprintf	sprintf_s
memcpy	memcpy_s
gets	gets_s

Coding - Validate!

- Validate Input and Output to Mitigate Common Vulnerabilities

"Checking the validity of incoming data and rejecting non-conformant data can remedy the most common vulnerabilities that lead to denial of service, data or code injection and misuse of end user data. In some cases, checking data validity is not a trivial exercise; however, it is fundamental to mitigating risks from common software vulnerabilities."

Secure Development

Coding

- Use Robust Integer Operations for Dynamic Memory Allocations and Array Offsets
  - Many memory corruption vulnerabilities today in C and C++ are due to incorrect math calculating dynamic memory sizes and array offsets

```
unsigned short x = 65535;
x++;
```

Secure Development

Coding

- Use Anti-Cross Site Scripting (XSS) Libraries
  - A focused subset of the “verify output” practice
  - All SAFECode members follow this pattern:
  - Constrain
  - Normalize
  - Validate
  - Encode

Secure Development

Coding

- Use Canonical Data Formats
- Avoid String Concatenation for Dynamic SQL Statements
- Eliminate Weak Crypto
- Use Logging and Tracing

Secure Development

Testing

- Determine Attack Surface
  - Understand what your software is exposing to the world
  - Local vs remote, anonymous vs authenticated
  - As important as getting the code right

> It should be stressed that testing is not a replacement for a development process that helps build more secure software, but rather that security testing is a core part of such a software development process.

Secure Development

Testing

- Fuzz Testing
  - If your software accepts input and you have never fuzz tested the app, you will find bugs!
  - Fuzz testing is “lying and cheating” about data
- Penetration testing
  - Often valuable, no replacement for a good process

Secure Development

Technology

- Use a current compiler suite
  - Vendors add defenses to the compiled code in new versions
- Use static analysis
  - We are big believers in static analysis, but …
  - “A fool with a tool is a still a fool!”

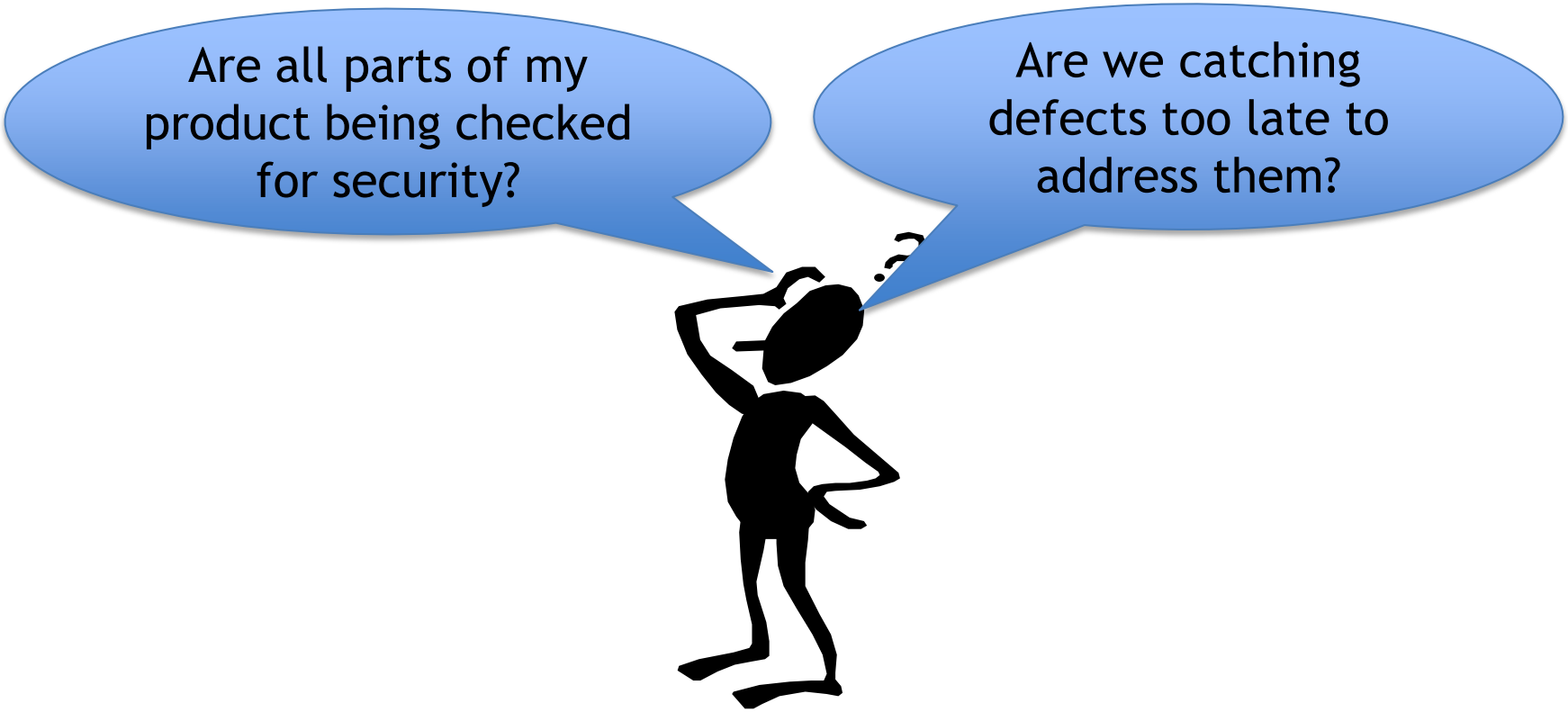
SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Secure Development Practices

Section	Practice
Secure Design Principles	Threat Modeling
	Use Least Privilege
	Implement Sandboxing
Secure Coding Practices	Minimize Use of Unsafe String and Buffer Functions
	Validate Input and Output to Mitigate Common Vulnerabilities
	Use Robust Integer Operations for Dynamic Memory Allocations and Array Offsets
	Use Anti-Cross Site Scripting (XSS) Libraries
	Use Canonical Data Formats
	Avoid String Concatenation for Dynamic SQL Statements
	Eliminate Weak Cryptography
	Use Logging and Tracing
Testing Recommendations	Determine Attack Surface
	Use Appropriate Testing Tools
	Perform Fuzz / Robustness Testing
	Perform Penetration Testing
Technology Recommendations	Use a Current Compiler Toolset
	Use Static Analysis Tools

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Thoughts to Consider



Do your suppliers and vendors take as much interest in security as you do?

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

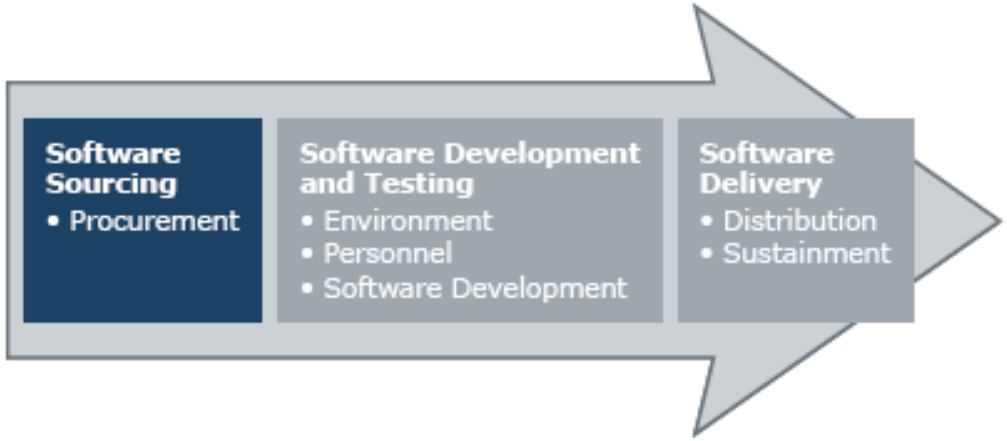
Supply Chain Integrity

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Supply Chain Integrity

Sourcing

- Security in contract language
- Vendor accountability
- Vendor security practices
- Intellectual property protection
- Secure storage and transfer
- Incident and Vulnerability response
- Malware identification



Supply Chain Integrity

A note about Open Source Software

- Contracts sometimes cannot be made between responsible parties
- Therefore, component selection criteria may be more stringent up front
  - Ensure the open source project has a strong, active community
  - Verify secure development and release engineering practices exist and are followed
  - Understand the goals of the community and how they support your needs
  - Don’t be afraid to contribute – enable security in the community

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Supply Chain Integrity

People, Process, Practice

- Clearly defined roles and responsibilities, with “sufficient” access rights
- Training
- Physical security
- Network and data segregation
- Secure repositories
- Secure builds
- Collaborative review and analysis

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Supply Chain Integrity

Delivery and Maintenance

- Malware checking
- Code signing
- Checksums / Hashes
- Secure distribution and hosting
- Runtime verification
- Reputable update notification
- Secure configurations
- Customization
- Pushing security updates

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Software Supply Chain Integrity Controls

Processes	Controls		
Software Sourcing	**Vendor Contractual Integrity Controls**	•Defined expectations •Ownership and responsibilities	•Vulnerability response •Security training
	Vendor Technical Integrity Controls for Suppliers	•Secure transfer •Sharing of system and network resources •Malware scanning	•Secure storage •Code exchange
Software Development & Testing	**Technical Controls**	•People security •Physical security •Network security	•Code repository security •Build environment security
	Security Testing Controls	•Peer review	•Testing for secure code
Software Delivery & Sustainment	**Publishing & Dissemination Controls**	•Malware scanning •Code signing	•Delivery •Transfer
	Authenticity Controls	•Cryptographic hashed or digitally signed components •Notification technology	•Authentic verification during program execution
	Product Deployment and Sustainment Controls	•Patching •Secure configurations	•Custom code extension

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Thoughts to Consider

Does "NIH" mean more time is spent reinventing the wheel?

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Summary / Wrap-Up

SAFECode SwA Lifecycle Practices shared by member companies

- Adobe
- EMC
- Juniper Networks
- Microsoft
- Nokia
- SAP
- Symantec

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

www.safecode.org
Twitter: @safecodeforum
Blog: http://blog.safecode.org

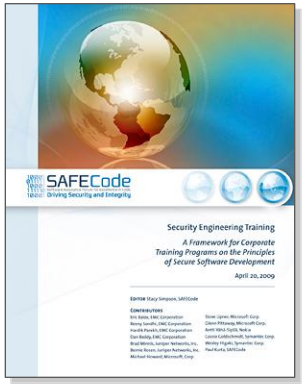
Stacy Simpson
SAFECode Director
703-812-9199
stacy@safecode.org

Security Engineering Training

Security Engineering Training: A Framework for Corporate Training Programs on the Principles of Secure Software Development

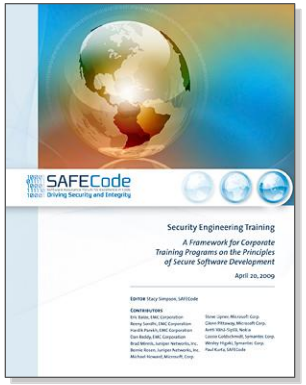
- **Focus:** Provide a framework for the development of corporate training programs on the principles of secure software development.

- **Key Objectives:** Assist others in the industry in developing their own security engineering training initiatives by offering insight into the common elements of training programs in place today within SAFECode member companies.



Security Engineering Training: A Framework for Corporate Training Programs on the Principles of Secure Software Development

- **Key Areas Covered:**
  - Creating a framework for internal security engineering training
  - Defining training targets and learning objectives
  - Developing or obtaining training content
  - Determining how to implement the training program



SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Secure Development Practices

Fundamental Practices for Secure Software Development – Second Edition

- **Focus:** Provide a foundational set of secure development practices based on an analysis of the real-world actions of SAFECode members

- **Key Objectives:** Help others initiate or improve their own software security programs and encourage the industry-wide adoption of fundamental secure development methods.

Secure Development Practices

Fundamental Practices for Secure Software Development – Second Edition

- **New** in 2nd Edition:
  - **Verification** methods and tools were developed for each listed practice to help managers confirm whether a practice was applied.
  - **Common Weakness Enumeration (CWE)** references were added to each practice to provide a more detailed illustration of the security issues these practices aim to resolve.

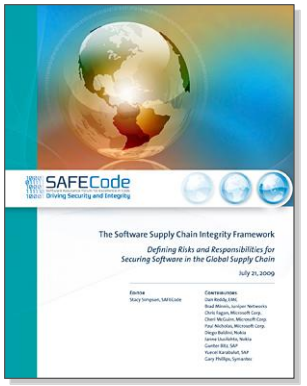
SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Software Supply Chain Integrity Framework

The Software Supply Chain Integrity Framework: Defining Risks and Responsibilities for Securing Software in the Global Supply Chain

- **Focus:** Provide the first industry-driven framework for analyzing and describing the efforts of software suppliers to mitigate the potential that software could be intentionally compromised during its sourcing, development or distribution.

- **Key Objectives:** Create a foundation for evaluating and describing software supply chain risks to enable the identification and analysis of mitigating controls and practices.



SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

Software Supply Chain Integrity Controls

Software Integrity Controls: An Assurance-Based Approach to Minimizing Risks in the Software Supply Chain

- **Focus:** Provide actionable recommendations for minimizing the risk of vulnerabilities being inserted into a software product during its sourcing, development and distribution.

- **Key Objectives:** Help others initiate or improve their software supply chain security programs and encourage broad industry adoption of software integrity controls.